MEASURING THE SIMILARITY OF TWO IMAGE SEQUENCES

Ying Shan, Harpreet S. Sawhney, Art Pope

Computer Vision Laboratory
Sarnoff Corporation
201 Washington Road, Princeton, NJ 08540, USA

ABSTRACT

We propose a novel similarity measure of two image sequences based on shapeme histograms. The idea of shapeme histogram has been used for single image/texture recognition, but is used here to solve the sequence-to-sequence matching problem. We develop techniques to represent each sequence as a set of shapeme histograms, which captures different variations of the object appearances within the sequence. These shapeme histograms are computed from the set of 2D invariant features that are stable across multiple images in the sequence, and therefore minimizes the effect of both background clutter, and 2D pose variations. We define sequence similarity measure as the similarity of the most similar pair of images from both sequences. This definition maximizes the chance of matching between two sequences of the same object, because it requires only part of the sequences being similar. We also introduce a weighting scheme to conduct an implicit feature selection process during the matching of two shapeme histograms. Experiments on clustering image sequences of tracked objects demonstrate the efficacy of the proposed method.

1. INTRODUCTION

The problem of matching two image sequences arises from the visual interpretation applications where a collection of images of each object is available. As compared with a single image, a sequence captures more appearance variability of the object, and is therefore more robust for the matching purpose. On the other hand, most low-level processes such as feature detection are not stable enough when operated on a single image. Having a sequence improves the chance of detecting stable features that are common to the images of the same object.

Figure 1 illustrates a typical example of our targeting problem set. The similarity of two sequences can be naturally defined as the similarity of the most similar pair of images from both sequences. In this case, A_7 (see Fig.1 for numbering details) and B_2 can be used to compute the similarity since they look the most similar. In reality, several issues need to be resolved before the similarity measure can actually be computed. First, each sequence may contain over hundreds of images, and hence exhaustively computing pairwise similarities is infeasible. Second, the effect of pose variations of the object need to be removed before the pairwise similarity measure is computed. Third, the background pixels such as in B_2 to B_7 need to be excluded from the computation of the pairwise similarities. Other difficulties such as illumination changes can also occur, but are not the concern of this paper.

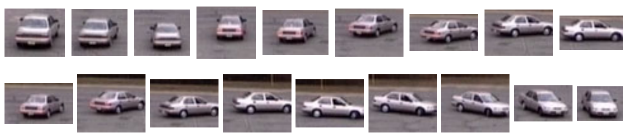

Fig. 1. Matching sequence A (above) against B (below). Images in each sequence can have different object appearances, 2D poses, and a fair amount of background clutter. The number and sizes of images in the sequences can be different, and the order of the images is of no importance. From left to right , the images are numbered from 1 to 9, and therefore A_1 represent the top left image from sequence A, and B_9 represents the bottom right image from sequence B.

To address these issues, our proposed method computes a feature vector for each image in the sequence. Clustering in the feature vector space generates a set of prototype feature vectors, which forms the representation of the sequence. The similarity between two sequences is then computed from their corresponding two sets of prototype feature vectors. Since the number of prototype feature vectors is usually much less than the original number of images in the sequence, the computational cost is largely reduced.

Our feature vector is essentially a histogram of low-level features that are affine invariant. Consequently, each prototype feature vector encodes information of a group of images that are similar up to an affine transformation. The appearance changes between different groups of images caused by, say, self-occlusion, can not be canceled by a simple 2D transformation. This is why we use multiple prototype feature vectors to represent a sequence. For example, we may need two prototype feature vectors to represent the group of images $\{B_4, B_5, B_6\}$, and $\{B_8, B_9\}$, respectively.

The set of all low-level features from the whole sequence is also clustered, and only those forming strong clusters in the feature space are selected and used in the binning of the histogram (the feature vector). This process intends to keep the foreground features and remove the background features, because the former is usually stable over the sequence, while the latter is not.

2. RELATED WORK

The idea of matching faces with two image sequences instead of two single images was presented in [1], where each sequence is represented as a set of straightened image vectors. The dissimilarity is then defined as the principle angle between the subspaces spanned by these two sets of vectors. While promising when applied to normalized face sequences, the underlying linear subspace assumption is somewhat restrictive. Obviously, images from sequence A or B in Fig.1 do not span a linear subspace even the 2D pose variations are canceled in advance. The Kernal Principal Angle [2] approach also uses the same dissimilarity measure, but maps the image vectors into a higher dimensional space before the principal angle is computed. The idea there is that a non-linear manifold in the image vector space can be represented as a linear subspace in the mapped higher dimensional space. This may be possible when the image vectors lay on a relatively low dimensional manifold induced by 3D pose variations, and limited global illumination changes. However, it will be interesting to see the performance when a fair amount of background pixels are included.

Another related approach is the part-based model representation and recognition approach[3], where scale invariant features are detected in the first place as the low-level features. Following [4], a probabilistic model with fixed number of parts is learnt from a set of images of a common object class, say, motorbike. Recognition is performed by assigning parts to different features and pick the hypothesis that gives the highest posterior. The recognition process has $O(M^P)$ complexity, where M is the number of feature in a single image, and P is the number of parts. This approach is therefore not appropriate for matching two sequences because the complexity will increase to $O(L * M^P)$, where L is the number of the images in the query sequence (the one that is not used to learn the model). Moreover, for the sequences given in Fig.1, multiple models are needed since one model can not explain all the appearance changes. This will further complicate the model estimation process. Instead of using a part-based probabilistic model, our approach essentially uses a set of averaged part histograms to describe the whole sequence. This representation captures more variability of the sequence, and is computationally more efficient.

Our basic representation is based on the idea of shapeme and shapeme histogram,proposed by Mori, Belongie and Malik [5] for 2D object recognition. A shapeme is a prototype shape feature that represents a cluster of similar invariant features called "2D shape contexts". Shape context features are computed based on basis points that are densely sampled on the 2D object, and hence a single object may contain many shape context features. Each feature on an object is assigned to its closest shapeme. Counting the frequency of shapemes over the whole object gives the shapeme histogram for the object. Fig. 2 shows an example. Twenty basis points are selected from a star-shaped object in Fig. 2(a). 2D shape features are computed and labeled by 3 shapemes as in Fig. 2(b). The shapeme labels are given in Fig. 2(c). Since there are 5 #1, 5 #2, and 10 #3 labels on the object, the shapeme histogram of this object is $\mathbf{h} = [5, 5, 10]$. Shapeme histogram is used by [6] and [7] for texture representation. In this paper, we use shapeme histogram in a different context, i.e., to represent a image sequence of an object.

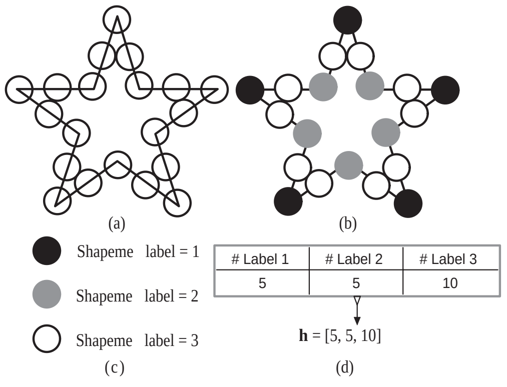

Fig. 2. Shapeme histogram. (a) The original 2D object, and the basis points (the circles) where 2D invariant features are computed. (b) The same object with each feature point labeled with a set of 3 shapemes labeled as in (c). (d) The shapeme histogram of the object in both table and vector formats.

3. OUR APPROACH

Given a sequence s that contains λ_s images, we compute one shapeme histogram $\mathbf{u}_k$ for each image. Clustering the set of shapeme histograms $\mathbf{U}^s = \{\mathbf{u}_1^s, \cdots, \mathbf{u}_{\lambda_s}^s\}$ generates a set of prototype shapeme histograms $\mathbf{V}^s = \{\mathbf{v}_1^s, \mathbf{v}_2^s, \cdots, \mathbf{v}_{\mu_s}^s\}$, where μ_s is the number of prototype shapeme histograms selected from $\mathbf{U}^s$. Each $\mathbf{v}_i^s$ encodes invariant appearance information of a group of images in the sequence that are similar up to an affine transformation.

Figure 3 shows the representative images for such groups from a sequence of 306 images. Because μ_s is usually much smaller than λ_s, the set $\mathbf{V}^s$ forms a compact representation of the sequence. The similarity of sequence p and sequence q, represented as $\mathbf{V}^p$ and $\mathbf{V}^q$, respectively, is then defined as:

$$\Psi(\mathbf{V}^p, \mathbf{V}^q) = \max_{i\in\{1,\cdots,\mu_p\}, j\in\{1,\cdots,\mu_q\}} \psi(\mathbf{v}_i^p, \mathbf{v}_j^q) \quad (1)$$

where $\psi(\cdot,\cdot)$ is the similarity between a pair of shapeme histograms.

3.1. Compute Prototype Shapeme Histograms

3.1.1. Local Process

Figure 4 shows how to compute $\mathbf{V}^s$ from a sequence s. The item number in the following discussion corresponds to the number in the figure. **1**. For each image in the sequence, a set of low-level features is extracted. **2**. K-Means clustering is then applied to find clusters in the feature space. Foreground features tend to have consistent appearance over multiple images in the sequence, and can therefore form strong clusters (dotted circles) in the feature space. **3**. These strong clusters are kept, while the weak clusters (the cyan points in Fig. 4) corresponding to unstable background features are removed. In our current implementation, clusters whose number of features are larger than a threshold τ are regarded as strong clusters. **4**. The centers of the remaining clusters are then used to generate the codebook, where each center is given a label. These centers are essentially the shapemes as described in Fig.2. **5 & 6**. The features in each image are then labeled according to the codebook, and by binning the labeled features as in Fig.2, a shapeme histogram $\mathbf{u}_k^s$ can be constructed for each image. **7**. Clustering for all the shapeme histograms of the sequence generates the set of prototype shapeme histograms $\mathbf{V}^s$. In the last step, images whose shapeme histograms are the closest to the prototype shapeme histograms are selected as the representative images. It should be noted that the representative images are the byproduct of this process. They are useful for the visualization purpose, but are not involved in the similarity computation.

While our shapeme histogram generation process seems similar to [5, 6, 7], it is important to note that our shapemes are computed from the features that are stable across multiple images. This is different from the single image/texture recognition applications, where shapemes are computed from the features that are spatially stable over a single image.

3.1.2. Global Process

The above process generates shapeme histogram based on a codebook that is local to each sequence. In order to compute similarity between different sequences, the actual shapeme histograms are computed from a unified global codebook. In our implementation, the global codebook is obtained by clustering all the stable features from all the sequences under consideration. The stable features are identified as the features belonging to the strong clusters that have been used to generate the local codebook.

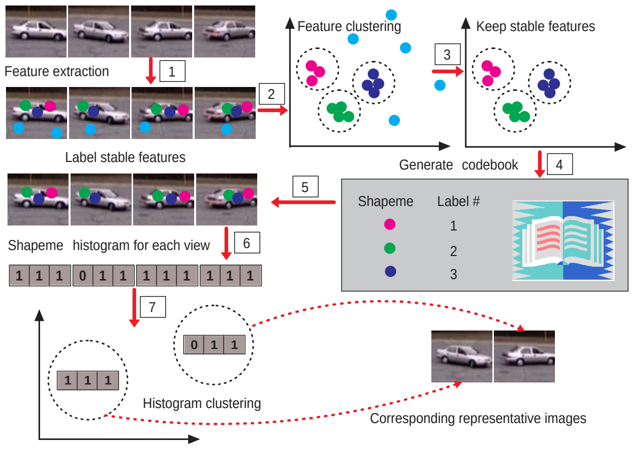

Fig. 4. Compute feature vectors and representative images.

3.2. Similarity of a Pair of Shapeme Histograms

Given a global codebook of κ shapemes, the similarity between two shapeme histograms $\mathbf{v_i}^p = [f_{i,1}^p, \cdots, f_{i,\kappa}^p]$ and $\mathbf{v_j}^q = [f_{j,1}^q, \cdots, f_{j,\kappa}^q]$ is defined as:

$$\psi(\mathbf{v}_i^p, \mathbf{v}_j^q) = \sum_{k=1}^{\kappa} w_{i,k}^p \cdot w_{j,k}^q \quad (2)$$

where $w_{i,k}^{\{p,q\}}$ is defined as:

$$w_{i,k}^{\{p,q\}} = \frac{f_{i,k}^{\{p,q\}} \log(N/n_k)}{\sqrt{\sum_{l=1}^{\kappa} (f_{i,l}^{\{p,q\}})^2 [\log(N/n_l)]^2}} \quad (3)$$

In (3), N is the total number of sequences under consideration, and n_k is the number of sequences that contain the kth shapeme in the codebook. This formula is known as $tf \times idf$ weight [8] in the document analysis community, where tf corresponds to $f_{i,k}$ called the *term frequency*, and *idf* corresponds to $\log(N/n_k)$ called *inverse document frequency*. A probabilistic justification can be found in [9]. It can be seen from both (2) and (3) that a high tf term can contribute to the decision that two histograms are similar. However, if the corresponding shapeme has low *idf* term, the contribution will be down weighted. This is essentially a feature selection process, where shapemes that belong to many sequences are regarded as less important. In addition

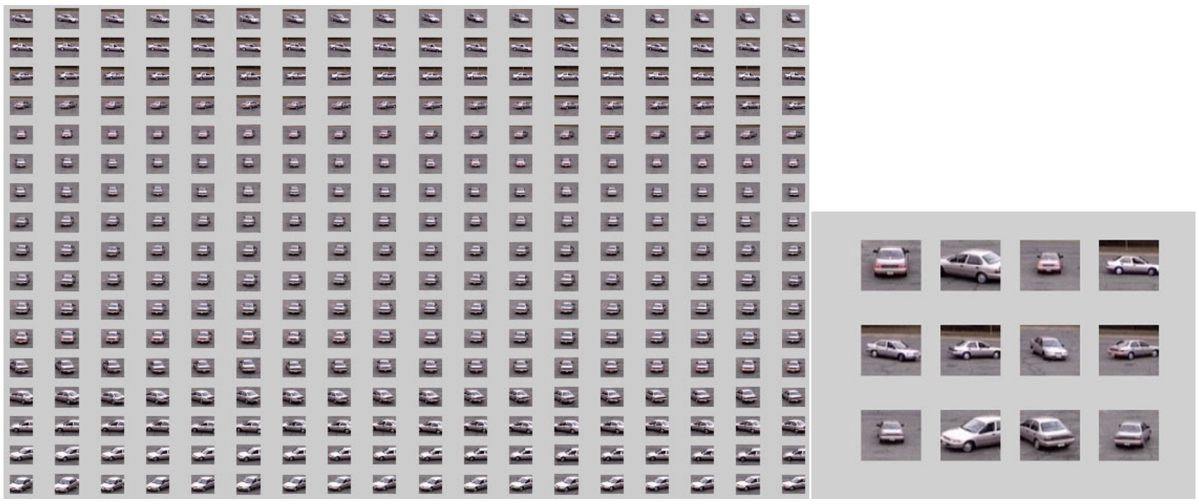

Fig. 3. Left: A sequence of 306 images. Right: Representative images of the sequence. Each image represents a group of images that are similar up to an affine transformation.

to tf and idf terms, the normalization term in (3) ensures that the similarity is not biased by the absolute number of features contained in each sequence. In other words, only the relative frequency of each shapeme is important in the comparison of two shapeme histograms.

3.3. Feature Extraction

The framework of our match measure computation does not depend on any particular type of low-level feature. Traditionally, such features are computed in the neighborhood of a point feature, and the size of the neighborhood is determined by a characteristic scale found by a scale space method[10, 11]. In our case, because the images are usually very small, e.g., less than 80×80 pixels, we find the intensity profile feature described in Fig. 5 gives better performance. A similar feature has been used in the context of image matching by Tell and Carlsson in [12], but is not invariant to the order of the two end points.

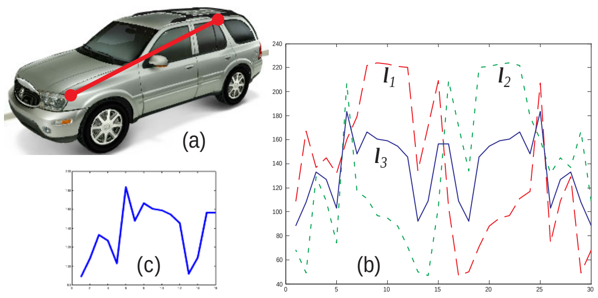

Fig. 5. Affine invariant profile feature computed from a pair of point features. Curve l_1 (dashed in red) in (b) corresponds to the intensity profile (left to right) along the curve that connects two point features in (a). Curve l_2 (dotted in green) in (b) is the reflection of the original intensity profile. Curve l_3 (solid in blue) is the average of the previous two curves. The solid curve in (c) corresponds to the left half of the averaged curve in (b). We use 10 samples from this curve to form a 10 dimensional feature vector, which is invariant to affine transformation, as well as the order of the two end points.

4. EXPERIMENTS

We have built on PC a prototype system that computes the proposed similarity measure for pairs of image sequences, and applied it for sequence clustering to demonstrate its efficacy. Sequences used in the experiments are prepared by cropping object images from their corresponding ROIs in the original videos. These ROIs are provided by a layer tracker for each tracked object. No other preprocessing such as alignment and background clutter removal are applied to the cropped images. Fig.6 shows some images of a vehicle generated in this way. It can be seen from Fig.6 that the images contain a fair amount of background clutter, significant appearance changes, and 2D pose variations. The experiments are conducted as the following.

1. For all the N sequences under concern, find all the point features for each image in the sequence, and compute profile feature as described in Sec. 3.3.

2. Compute the set of shapeme histograms $\mathbf{V}^s$ for each sequence s as described in Sec. 3.1. Note here that the codebook is global to all the sequences.

3. Compute similarity measures between pairs of sequences as describe in Sec. 3.2.

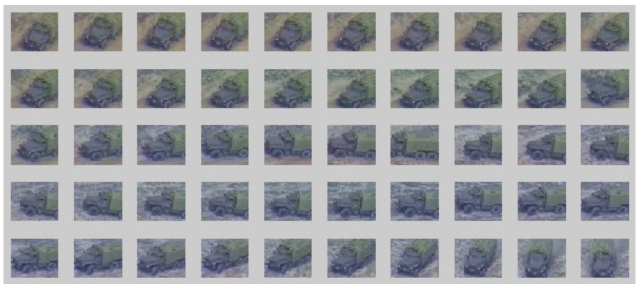

Fig. 6. Fifty vehicle images from a sequence of 120 images.

4. Convert similarity measure into dissimilarity measure, and form a $N \times N$ dissimilarity matrix for all the N sequences. Since our similarity measure $\Psi \in [0, 1]$, we simply use $1 - \Psi$ to convert it into a dissimilarity measure.

5. Visualize the dissimilarity matrix with dendrogram.

There are four parameters to adjust, i.e., the number μ of the shapeme histograms for each sequence, the number ν of the clusters used to generate the stable features as in Fig. 4, the number κ of the clusters used to generate the global codebook as described in Sec. 3.1, and the threshold τ for defining the strong clusters as described in Sec. 3.1. After some experiments, the fixed set of parameters, i.e., $\mu = 5$, $\nu = 100$, $\kappa = 500$, $\tau = 10$, is used for all the experiments presented here. The number of point features used to generate the profile features is limited to 30 per image.

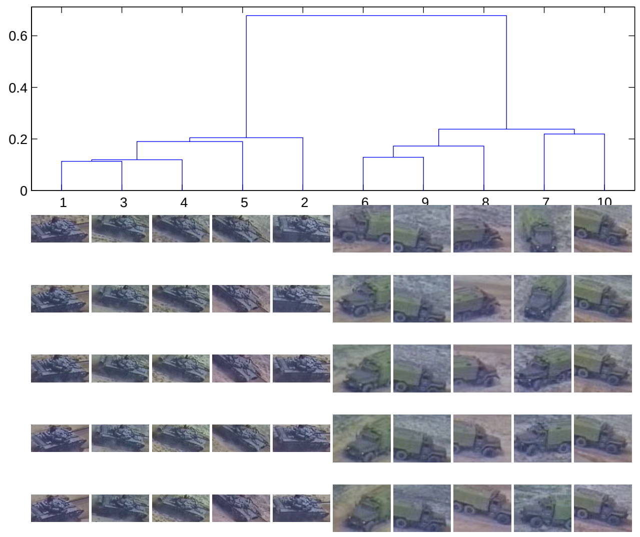

Fig. 7. Clustering result for 10 sequences of 2 vehicle objects.

The first experiments uses 10 sequences of 2 vehicle objects, five sequences for each object, and about 120 images for each sequence. These sequences are taken from a moving aerial camera looking at moving objects on the ground. Figure 7 shows the result visualized by a dendrogram, which is plotted by a modified Matlab function called "dendrogram". Dendrogram is a tree structure, of which each leaf node represents a sequence in our case. For each sequence, we use $\mu = 5$ representative images displayed in a column in the dendrogram. These representative images are generated as described in Sec. 3.1 and Fig. 4. The vertical axis on the top part of the dendrogram shows the dissimilarity values. The smaller this number, the more similar the two sequences are. Similarly, the further (in terms of tree levels) a tree node is from the root, the more the groups of sequences (under this node) are similar.

It can be seen from Fig. 7 that 10 sequences of the two objects are clearly separated into two corresponding groups, one for the sequences of $\{1, 3, 4, 5, 2\}$ (the sequence numbers are on the horizontal axis), and the other for $\{6, 9, 8, 7, 10\}$. The dissimilarity between these two groups are very high (above 0.6), while the dissimilarities within each group are much smaller (below 0.25). The 5 representative images from the 7th sequence in Fig. 7 are selected automatically from the sequence shown in Fig. 6. It can be seen that the system indeed captures the major appearance changes in the sequence. The foreground objects in some other sequences such as sequences $\{1, 5, 3\}$ do not have much appearance variances, and hence their representative images look all the same. The running time is about 11 minutes on a 850Hz PC. Our second experiment uses 12 sequences of 5 vehicle objects, with 3 of them having similar shape and color. These sequences are taken from a static camera looking from the side of a circular drive way. Each vehicle is driven in front of the camera at least twice in alternative directions. The result is given in Fig. 8. According to the truth that $\{4, 7\}$, $\{2, 8\}$, $\{3, 11, 9\}$, $\{5, 12\}$, and $\{1, 6, 10\}$ belong to the same object, the only confusion is caused by the 9th sequence, which is quite different from $\{3, 11\}$ due to the out-of-plane rotation of the vehicle. However, it should be noted that the 9th sequence is still closer to $\{3, 11\}$ than $\{5, 12\}$, and $\{1, 6, 10\}$. Also note that the scale change between $\{5\}$ and $\{12\}$ is properly handled. The reflection invariance seen as in $\{4, 7\}$ and $\{2, 8\}$ is due to the fact that our profile feature is invariant to the order of the end points. The running time is about 15 minutes for this experiment.

5. CONCLUSION AND FUTURE WORK

We introduce a novel idea of using shapeme histogram to solve the sequence-to-sequence matching problem, which is challenging in the presence of large appearance and pose variations, and background clutter. We propose to represent each sequence as a set of prototype shapeme histograms, each of which encodes information of a group of

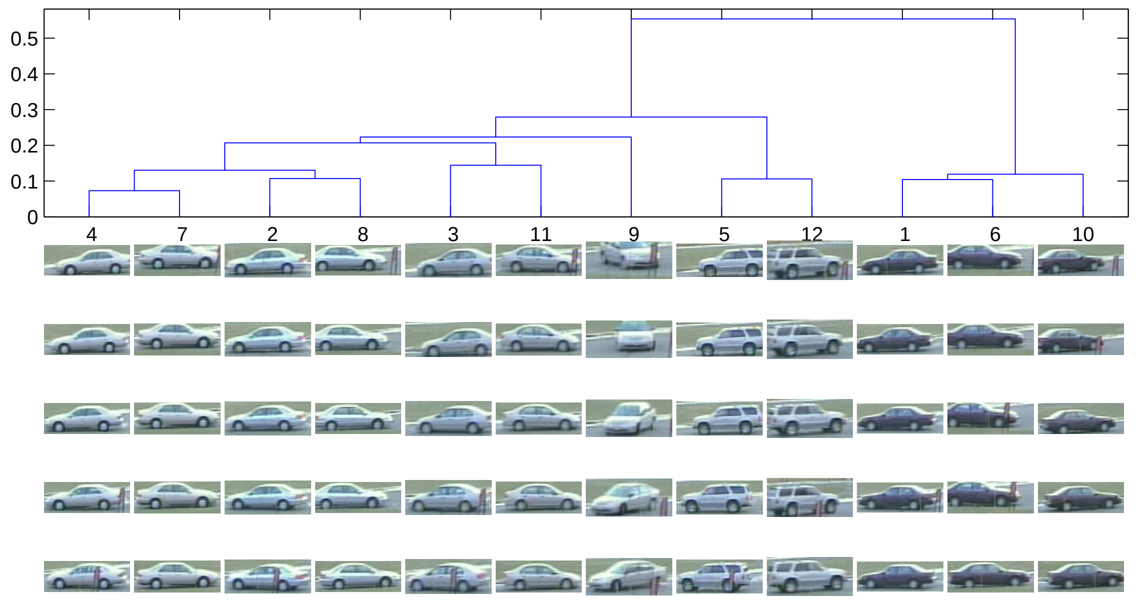

Fig. 8. Clustering result for 15 sequences of 6 vehicle objects.

images with similar appearance. This representation captures the major appearance changes in the sequence while still keeps the matching computation tractable. We develop techniques to compute shapeme histograms from the set of affine-invariant features that are stable across multiple images in the sequence, and therefore reduce the difficulty caused by both background clutter and affine pose variations. We also borrow the $tf \times idf$ weighting scheme to conduct an implicit feature selection process when matching two shapeme histograms. Experiments on clustering image sequences of tracked objects confirms the efficacy of the proposed method.

The current approach uses intensity profile feature, which is not invariant to illumination changes, and is computationally expensive. In the future we will look for new features that can be easily extracted from small images, and are invariant to both pose and illumination changes. Instead of having a fixed number of prototype shapeme histograms for all the sequences, it will be also interesting to find an optimal number for each sequence.

6. REFERENCES

[1] Yamaguchi O., Fukui K., and Maeda K., "Face recognition using temporal image sequence," in *International Conference on Automatic Face and Gesture Recognition*, 1998, pp. 318–323.

[2] Lior Wolf and Amnon Shashua, "Kernel principal angles for classification machines with applications to image sequence interpretation," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR03)*, 2003.

[3] R. Fergus, P. Perona, and A. Zisserman1, "Object class recognition by unsupervised scale-invariant learning," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR03)*, 2003.

[4] M. Weber, M. Welling, and P. Perona, "Unsupervised learning of models for recognition," in *European Conference on Computer Vision (ECCV00)*, 2000, pp. 18–32.

[5] G. Mori, S. Belongie, and J. Malik., "Shape contexts enable efficient retrieval of similar shapes," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR01)*, 2001, pp. 723–730.

[6] Thomas Leung and Jitendra Malik, "Representing and recognizing the visual appearance of materials using three-dimensional textons," *International Journal of Computer Vision (IJCV01)*, vol. 43, no. 1, pp. 29–44, 2001.

[7] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce, "A sparse texture representation using affine-invariant regions," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR03)*, 2003.

[8] Gerard Salton, "Developments in automatic text retrieval," *Science*, vol. 253, no. 5023, pp. 974–980, 1991.

[9] Djoerd Hiemstra, "A probabilistic justification for using tf × idf term weighting in information retrieval," *International Journal of Digital Library*, vol. 3, pp. 131–139, 2000.

[10] K.Mikolajczyk and C.Schmid, "An affine invariant interest point detector," in *European Conference on Computer Vision (ECCV02)*, 2002, vol. 4, pp. 700–714.

[11] David G. Lowe, "Object recognition from local scale-invariant features," in *International Conference on Computer Vision(ICCV99)*, 1999, pp. 1150–1157.

[12] Dennis Tell and Stefan Carlsson, "Combining topology and appearance for wide baseline matching," in *European Conference on Computer Vision (ECCV02)*, 2002, pp. 68–81.